

Broadview
www.broadview.com.cn

Zhou Zhangfa's Series
周长发作品系列

.NET
技术大系

C# 面向对象编程

周长发 编著

以**面向对象编程**和**.NET框架类库**为主轴，由知名技术作家基于**C#最新版本**而作，是全面详细的**C#技术宝典**。

光盘中包含书中全部源代码、示例程序的可执行文件、相关文件。



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

 Zhou Zhangfa's Series
周长发作品系列



C# 面向对象编程

周长发 编著

電子工業出版社

Publishing House of Electronics Industry

北京•BEIJING

内 容 简 介

本书以面向对象编程技术和.NET 框架类库为主轴,透彻地介绍了 C#的语法、编程思想和高级.NET 框架编程技术。全书分为三部分,共 22 章,全面详细地解说了 C#的技术要点,包括 C#语言基础、C#的面向对象特征、异常处理、委托与事件、集合、泛型、字符串与正则表达式、特性、反射、程序集与版本控制、.NET 框架的内存管理、多线程、Windows 窗体、GDI+、与非托管代码互操作等,对每一个知识点都提供了典型实例。本书基于 C#和.NET 框架的最新版本而编写,介绍了 C#的最新特色,包括泛型、匿名方法、迭代器、静态类、分部类型和可空类型等。本书既阐述了 C#的基础知识,又介绍了 C#的高级应用技术,包括多线程、异步调用、图形图像处理、游戏编程、与 COM 等非托管代码的互操作,以及插件编程等。

本书适合所有希望了解和掌握 C#和.NET 的开发人员阅读,是大专院校计算机相关专业和各类培训班的理想教学参考书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

C#面向对象编程 / 周长发编著. —北京: 电子工业出版社, 2007.7

(.NET 技术大系)

ISBN 978-7-121-03813-6

I. C… II. 周… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2007)第 010839 号

责任编辑: 高洪霞

印 刷: 北京东光印刷厂

装 订: 三河市皇庄路通装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 49.25 字数: 1218 千字

印 次: 2007 年 7 月第 1 次印刷

印 数: 5000 册 定价: 89.80 元(含光盘 1 张)

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zltts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。



作者简介

周长发，北京大学理学博士，现在美国硅谷一家软件公司工作。他的主要研究领域为图像处理、多媒体技术、软件设计与架构以及计算机安全。他撰写了《精通Visual C++图像处理编程》和《科学与工程数值计算算法集（Visual C++版）》等8本书籍，还翻译了《计算机图形学几何工具算法详解》和《黑客调试技术揭秘》两本专著。

前 言

C#是一种简洁、类型安全的面向对象编程语言，可以用于构建在 .NET 框架上运行的各种安全、可靠的应用程序，包括传统的 Windows 客户端应用程序、XML Web 服务、分布式组件、客户-服务器应用程序，以及数据库应用程序等类型的程序。C#与 Java 非常类似，也是从 C/C++ 发展而来的一种面向对象的语言。对于熟悉 C/C++ 或者 Java 语言的程序员来说，只需要花费很少的时间就能掌握 C#。

C#剔除了 C++ 中繁杂和易于出错的一些规则，其语法非常简洁易用。在简化语法和提供类型安全的同时，C#还提供了一些比 C/C++ 和 Java 更强大的功能，包括特性 (attribute)、属性 (property)、可空类型 (nullable type)、枚举 (enum)、委托 (delegation)、匿名方法和直接内存访问等。C#是一种完全、纯粹的面向对象语言，支持封装、继承和多态性；C#程序的所有变量和方法，包括 Main 方法（应用程序的入口点），都封装在类定义中；C#的类只能从一个直接基类继承，但是类可以实现任意数量的接口；重写父类的虚方法时，必须使用 override 关键字，可以有效地避免意外的错误；C#支持泛型，提供了优秀的类型安全和性能；C#还提供了迭代器，允许集合类的实现者定义自定义的迭代行为，简化了客户端代码对它的使用。

.NET 已经成为 Microsoft 的支撑战略，Microsoft 的所有产品现在都围绕.NET 而展开。C#是.NET 的核心——.NET 框架——的“母语”，因此使用 C#能够最有效地开发基于.NET 的应用程序。实际上，Microsoft 早已开始使用 C#来开发其绝大部分新产品。

本书基于 C# 2.0 规范而著，融会了笔者十多年的专业编程经验。本书的目标是让读者理解 C#的编程思想，帮助读者掌握 C#这门优雅的语言，早日成为.NET 编程高手。

本书的适用对象

本书面向所有希望了解和掌握 C#和.NET 的开发人员。本书从最基本的 C#语法开始，说明了 C#语言的技术特征，以面向对象编程思想和.NET 类库为主轴，结合大量的典型实例，全面透彻地介绍了 C# 2.0 的编程方法和进行.NET 框架编程的高级技术。因此，本书适合于各个层次读者的需求，无论你是刚开始接触 C#，还是已有一定的 C#工作经验，都能从书中找到自己所需要的内容。

本书虽然并未预设任何前提，但是，笔者假定读者都具备计算机软件和编程的基础知识（比如，知道什么是内存、什么是程序等）。这类知识将有助于读者理解本书的内容。

本书的主要内容

为使读者理解 C#的面向对象编程思想，本书的内容是围绕“如何使用面向对象的 C#来进行思考”这一主旨而展开的。全书分为三个部分，共 22 章。在介绍了 C#的语法细节的基础上，本书阐述了 C#的面向对象思想，讨论了.NET 框架系统编程的高级技术。

第 1 部分，**C#语言基础**，介绍了 C#的基本语言特征。这一部分主要帮助读者熟悉 C#的基

本编程元素。如果读者熟悉 C/C++ 或 Java, 将会发现这一部分介绍的绝大部分内容都似曾相识, 可以快速地浏览这一部分内容。如果读者不太熟悉 C/C++ 或 Java, 或者没有太多的编程经验, 那么仔细阅读这一部分将有助于熟悉计算机编程语言的基本概念和 C# 的基本概貌。

第 1 章, **C#概述**, 介绍了 C# 的历史、特征和功能, 以及与 .NET 的关系, 阐述了 C# 编程的两个要点, 即其编程方法学为面向对象编程, 其编程库为 .NET 框架类库。本章还介绍了进行 C# 开发所需的基本工具。本章将帮助读者建立关于 C# 编程的全景图, 笔者建议所有的读者都仔细地阅读本章。

第 2 章, **开始 C#编程**, 介绍了 C# 程序的基本结构和编译方法; 简要说明了如何使用 Visual Studio 2005 进行软件开发和调试。

第 3 章, **变量与常量**, 介绍了计算机编程语言的基本概念——变量、常量、类型、类型转换, 以及这些概念在 C# 中的表现形式。本章还详细阐述了 C# 特有的装箱/拆箱操作和可空类型的概念。

第 4 章, **表达式与运算符**, 介绍了计算机编程语言的基本概念——表达式和运算符, 以及 C# 支持的运算符和它们的优先级。

第 5 章, **程序流程控制**, 说明了计算机编程语言的基本概念——语句、语句块和流程控制语句, 以及 C# 支持的选择、循环和跳转语句的功能和语法。

第 2 部分, **C#面向对象编程**, 介绍了 C# 的面向对象编程技术。在说明了面向对象编程技术的概念和方法的基础上, 这一部分详细阐述了 C# 的面向对象思想。这一部分内容是本书的核心, 值得仔细体会。

第 6 章, **面向对象编程技术**, 首先介绍了面向对象编程中经常使用的 UML, 然后重点阐述了基于对象建模技术的面向对象分析和面向对象设计的概念和方法。

第 7 章, **类: 抽象与封装**, 详细说明了如何通过概念抽象和数据封装来设计类, 如何在 C# 中声明类和类的成员。本章还介绍了 C# 特有的几个概念, 包括属性、索引器、命名空间和分部级。本章是理解 C# 面向对象编程的基础, 在进入下一章的学习之前, 请确保你已经完全理解本章的所有概念。

第 8 章, **方法**, 介绍了 C# 方法的声明方式、签名和重载, 作为程序入口点的 Main 方法, 以及递归方法。本章还提供了一个综合示例, 说明了如何基于面向对象的分析和设计来进行面向对象编程。

第 9 章, **继承与多态**, 论述了 C# 如何实现面向对象编程中的这两个核心原则, 介绍了抽象类、静态类、接口和组件编程的概念和方法, 并说明了类的转换方式。本章是理解 C# 面向对象编程思想的核心, 笔者期望读者能够理解本章的每一个概念。

第 10 章, **异常处理**, 介绍了 C# 错误处理的基本方法——异常处理。

第 11 章, **重载运算符**, 说明了 C# 重载运算符的原则和语法。

第 12 章, **结构与枚举**, 介绍了如何在 C# 中定义和使用结构与枚举。结构与枚举是两种值类型。

第 13 章, **集合**, 讨论了在 C# 中实现和使用集合的方法与技巧。与其他语言相比, C# 的数组是一种集合, 功能更强大, 更易于使用。本章介绍了 .NET 框架类库提供的几种常用的集合

类。使用迭代器能够非常简单地实现自定义集合类。

第 14 章，**字符串与正则表达式**，介绍了 C# 进行字符串处理的基本方法，以及如何使用正则表达式获取强大的字符串处理能力。

第 15 章，**委托与事件**，介绍了 C# 独创的委托概念，以及 C# 基于委托来实现其事件模型的方法。委托是一种用来存储方法引用的引用类型。事件是一种进行类间通信的函数成员，委托是定义事件的基础。委托与事件是深入理解 C# 编程思想的重要概念。

第 16 章，**泛型**，说明了 C# 如何通过泛型来提供安全和高效的类型支持，介绍了使用泛型集合的编程方法。

第 3 部分，**使用 .NET 框架类库**，说明了如何通过 C# 来使用 .NET 框架类库编程。.NET 框架类库是 C# 语言的编程库，C# 程序要实现任何有意义的功能，都必须通过调用 .NET 框架类库中定义的类来实现。另一方面，C# 是进行 .NET 框架编程的最佳语言，使用 C# 可以实现 .NET 框架中的任何功能。本章讨论了 .NET 框架系统编程的许多高级技术。对于具备一定的 C# 编程经验的读者来说，这一部分可能是他们最感兴趣的内容。

第 17 章，**文件操作**，介绍了 C# 语言在 .NET 系统中进行文件操作的方法，包括针对文本文件、二进制文件和 XML 文件的各种编程技术。

第 18 章，**特性、反射与程序集**，说明了 C# 如何使用特性和自定义特性，如何编写程序集和进行程序集版本控制，如何使用反射，以及 .NET 框架的自动内存管理方式。本章讨论的动态代码生成和插件编程技术可能是程度较高的读者所企盼的内容。

第 19 章，**多线程**，介绍了 .NET 框架的多线程管理和同步编程技术。本章还讨论了异步调用，这是一种非常有用的编程技术。

第 20 章，**Windows 窗体编程**，说明了如何编写 Windows 窗体应用程序，介绍了几种典型的常用控件。这是任何一个 C# 程序员都会感兴趣的内容。本章还讨论了用户界面与多线程的关系，理解这一点是编写高质量商业应用程序的基础。

第 21 章，**GDI+ 图形图像编程**，介绍了使用 GDI+ 来绘制和打印图形、图像和文本的方法，讨论了基于 GDI+ 的图像处理的技术，说明了如何编写和使用自定义控件。本章的综合示例实现了 Hanoi 塔游戏，其中介绍了许多用 C# 来编写游戏的技巧。

第 22 章，**与非托管代码的互操作编程**，说明了如何在托管代码中调用 DLL 中的非托管函数，如何在托管代码中使用 COM 组件和 ActiveX 控件，如何编写能够用于非托管代码的 .NET 程序集，如何在非托管代码中调用 .NET 程序集，以及如何在 C# 中使用不安全的类型（指针）。本章提供了一个使用 C# 进行底层图像处理操作的例子，介绍了提高 C# 程序的运行速度的方法。

本书还包括两个附录，列出了 C# 关键字，介绍了 C# 编程风格。附录 B 中的 C# 编程风格提供了一种实用的简明编程风格，有助于编写高质量的 C# 代码。

C# 是一种非常强大的编程语言，除了本书介绍的内容外，C# 还被广泛地应用于多媒体、安全与加密、数据库 (ADO.NET)、Web (ASP.NET)、Web 服务、网络访问与 .NET Remoting 等方面的编程。限于篇幅，不可能在本书中包含这些内容，但是它们可能是笔者下一本书的主要内容。

如何使用本书

本书提供了大量的实用代码和例程。在阅读本书时，读者最好能准备好一台计算机，以便随时能边读边练，结合本书提供的实例来学习。

为了能够正确地使用本书提供的示例程序，读者的计算机系统应满足如下基本要求。

- CPU：奔腾 III 或更高
- 内存：不少于 128MB
- 操作系统：Windows XP（Service Pack 2）
- 编程环境：Visual Studio 2005/.NET 框架 SDK 2.0

本书采用的术语翻译

本书尽量使用流行的术语翻译。但是，几个术语的流行翻译很容易引起误解，因此，笔者采用了笔者认为更准确的翻译。下表列出了这些术语的原文，本书采用的翻译，流行的翻译，以及不采用流行的翻译的原因：

原 文	本书采用的翻译	流行的翻译	不采用流行的翻译的原因
attribute	特性	属性	property 已经被译为“属性”。为了避免混淆 attribute 和 property，将 attribute 译为特性
enumerate	列举	枚举	enum 已经被译为“枚举”。为了避免混淆 enum 和 enumerate，将 enumarate 译为列举
enumerator	列举器	枚举器	保持与 enumerate 的翻译一致
override	重写	覆盖	“重写”更准确地表达了 override 的含义

本书的语法版式

本书在介绍 C#语法时，使用了灰色背景。语法中以“[”开始和“]”结束的汉语词汇（或者标点符号）说明可选的成分；以“【”开始和“】”结束的汉语词汇用来说明必需的成分。它们都是表意的，在程序中会被标识符替代。语法中出现的其他英语词汇和标点符号都是必需的，而且必须原样出现在程序中。比如，下面是用于声明类的语法：

```
【特性】
【类修饰符】 class 【类名】 [：【基类或接口】]
{
    【类体】
} [；]
```

说明在声明类时，【特性】是可选的；【类修饰符】也是可选的；关键字 class 和【类名】是必需的；基类或接口部分是可选的，但是如果存在，则【基类或接口】是必需的；【类体】是必需的，类体的代码包括在一对大括号之间；最后还可添加一个分号，这也是可选的。在程序中，所有以“[”开始和“]”结束，以及以“【”开始和“】”结束的部分都会被标识符替代。比如，基于上述的语法说明，下面的类声明代码都是有效的：

```

public abstract class Shape
{
    // 省略类体代码
}

class Rectangle : Shape
{
    // 省略类体代码
};

public class Ellipse : Shape
{
    // 省略类体代码
};

[Obsolete("该类已经过时，请避免使用。应该使用 Ellipse 类。")]
public class Circle : Shape
{
    // 省略类体代码
}

protected sealed class EmptyClass
{
    // 类体可以为空
}

```

本书的源代码

本书随附的光盘中包含了书中所有示例程序的源代码。

本书的许多示例都没有采用 Visual Studio 2005 的项目管理，而是直接编写代码，用 csc 命令行来编译。这样做的目的和好处如下：

(1) 突出 C# 程序的本质和编程思想。项目管理能够简化操作，但是同时也往往会掩盖事情的本质。

(2) 减小篇幅。直接编码只需要列出代码本身，使用项目管理则需要多费笔墨来介绍许多与代码无关的操作。

当然，在实际开发中，最好还是使用 Visual Studio 2005 的项目管理。本书也提供了一些这样的示例。

在本书给出的源代码中，如果代码的第一行没有使用注释列出文件名，那么代码是不完整的，只是代码片断。

如果代码的第一行使用注释列出了代码的文件名（比如，// Hello.cs），则说明代码是完整的程序。可以在代码包中，到与代码所在的章对应的目录下，找到该程序的源文件和编译好的可执行文件或程序集。比如，本书的第一个程序示例为第 2 章的“Hello World!”程序，它的代码的第一行为：

```
// Hello.cs
```

因此，可以在本书配套光盘的 .\CSharp\Chapter2 目录中找到其源文件 Hello.cs 和已经编译

好的可执行文件 Hello.exe。

使用 Visual Studio 2005 生成的项目都提供了源代码，保存在所在章对应的目录的一个名称为项目名的子目录中。比如，第 2 章的 HelloWorld 项目保存在 \CSharp\Chapter2\HelloWorld 目录中。

如何发表意见和获得技术支持

笔者欢迎读者提出宝贵意见。读者在阅读本书时遇到任何问题，或者发现了本书存在的错误或纰漏（限于笔者的能力，本书中出现错误、浅陋和陈旧之处在所难免），都请直率地告诉笔者。笔者会通过电子邮件尽力回复读者提出的问题。请通过如下电子邮件地址与笔者联系：

darin.zhou@gmail.com

致谢

感谢所有对本书的写作有过帮助的人。周建欣、林琳、黄家辉、邓优、李秀鹏、刘烁和何西涛等参加了本书有关内容的讨论和资料收集，周建欣、彭文、黄娅丽测试了本书的所有示例程序。

我在此特别感谢本书的责任编辑高洪霞女士，她细致而高效的编辑是本书得以快速出版的基础。我在此也特别感谢电子工业出版社博文视点资讯有限公司为本书的出版付出的努力。

周长发

2006 年 11 月

目 录

第 1 部分 C#语言基础

第 1 章 C#概述	2
1.1 C#溯源	2
1.2 C#与 Microsoft .NET	4
1.3 C#的基本特征	5
1.4 C#编程的两个要点	7
1.4.1 编程方法学：面向对象编程	7
1.4.2 编程库：.NET 框架类库	9
1.5 安装 C#开发工具	10
1.6 小结	11
1.7 练习	11
第 2 章 开始 C#编程	12
2.1 编写“Hello World!”程序	12
2.2 编译和执行程序	13
2.3 程序结构	14
2.3.1 注释	15
2.3.2 命名空间	15
2.3.3 类	15
2.3.4 Main()方法	15
2.3.5 方法调用	16
2.3.6 文本、标识符与关键字	16
2.3.7 语句	17
2.3.8 空白字符	17
2.4 使用 Visual Studio 2005 来编程	17
2.4.1 Visual Studio 2005 是一种集成开发环境	17
2.4.2 基于 Visual Studio 2005 的 C#编程	18
2.4.3 解决方案与项目	20
2.4.4 类视图	20
2.4.5 功能强大的文本编辑器	22
2.4.6 自动生成的代码分析	24

2.4.7	生成与查错	25
2.4.8	执行与调试	27
2.5	小结	29
2.6	练习	29
第3章	变量与常量	30
3.1	C#程序的内存类型	30
3.1.1	堆栈	30
3.1.2	托管堆	31
3.1.3	非托管堆	32
3.2	变量	32
3.3	类型	32
3.3.1	所有类型都是类	32
3.3.2	引用类型	33
3.3.3	值类型	33
3.3.4	基本类型	34
3.4	变量操作	38
3.4.1	声明变量和变量的作用域	38
3.4.2	给变量赋值	40
3.5	常量	41
3.6	类型转换	42
3.6.1	隐式转换	42
3.6.2	显式转换	44
3.6.3	装箱与拆箱	46
3.7	数组	48
3.8	可空类型	50
3.8.1	定义	51
3.8.2	包装与解包	51
3.8.3	可空转换	51
3.8.4	装箱与拆箱	52
3.9	三个基本的.NET 框架类	53
3.9.1	Console 类	53
3.9.2	Convert 类	57
3.9.3	Math 类	58
3.10	小结	59
3.11	练习	60

第 4 章 表达式与运算符	61
4.1 表达式	61
4.2 运算符	61
4.2.1 基本运算符	62
4.2.2 赋值运算符	65
4.2.3 算术运算符	65
4.2.4 一元加、减运算符	66
4.2.5 比较运算符	66
4.2.6 逻辑运算符	67
4.2.7 移位运算符	70
4.2.8 条件逻辑运算符	71
4.2.9 三元运算符	72
4.2.10 提升运算符	72
4.2.11 空合并运算符	74
4.3 运算符的优先级和结合性	75
4.4 小结	76
4.5 练习	76
第 5 章 程序流程控制	77
5.1 语句和语句块	77
5.2 选择语句	78
5.2.1 if...else	78
5.2.2 switch	81
5.3 循环语句	83
5.3.1 while	83
5.3.2 do...while	85
5.3.3 for	87
5.4 跳转语句	89
5.4.1 break	89
5.4.2 continue	90
5.4.3 return	90
5.4.4 goto	91
5.5 小结	93
5.6 练习	93

第 2 部分 C#面向对象编程

第 6 章 面向对象编程技术	96
6.1 UML 基础	96
6.1.1 简介	97
6.1.2 用例图	97
6.1.3 类图和对象图	98
6.1.4 序列图	102
6.1.5 状态图	103
6.1.6 活动图	103
6.1.7 组件图	104
6.1.8 部署图	104
6.2 对象建模技术	105
6.2.1 功能模型	106
6.2.2 对象模型	106
6.2.3 动态模型	106
6.3 面向对象分析	107
6.3.1 原则	107
6.3.2 阶段	108
6.3.3 目标	109
6.3.4 步骤	109
6.3.5 优点	110
6.4 面向对象设计	110
6.4.1 概念	110
6.4.2 原则	111
6.4.3 任务	111
6.4.4 阶段	112
6.4.5 过程和步骤	112
6.5 小结	114
6.6 练习	114
第 7 章 类：抽象与封装	115
7.1 声明类	115
7.2 类的成员	116
7.2.1 数据成员	116
7.2.2 函数成员	117

7.3	访问修饰符	118
7.3.1	类的成员的访问修饰符	118
7.3.2	类的访问修饰符	119
7.4	创建对象	120
7.5	静态成员与实例成员	122
7.6	构造函数与析构函数	124
7.6.1	构造函数	124
7.6.2	析构函数	129
7.7	只读字段	131
7.8	关键字 <code>this</code>	132
7.8.1	<code>this</code> 访问	132
7.8.2	<code>this</code> 用于构造函数声明	132
7.9	属性与索引器	133
7.9.1	属性	133
7.9.2	索引器	136
7.10	命名空间	139
7.10.1	编译单元与命名空间	139
7.10.2	完全限定名	140
7.10.3	<code>using</code> 指令	141
7.11	内部类	145
7.12	分部类	147
7.13	小结	149
7.14	练习	150
第 8 章	方法	151
8.1	声明方法	151
8.2	静态方法与实例方法	154
8.3	方法参数	156
8.3.1	值参数	157
8.3.2	引用参数	158
8.3.3	输出参数	158
8.3.4	参数数组	160
8.3.5	数组参数	161
8.4	方法重载	163
8.5	<code>Main</code> 方法	170
8.6	递归	172

8.7	综合示例：自动取款机模拟程序	175
8.7.1	需求	175
8.7.2	分析	175
8.7.3	设计	180
8.7.4	实现	180
8.8	小结	189
8.9	练习	189
第9章	继承与多态	190
9.1	继承	190
9.1.1	定义类的层次结构和基类	190
9.1.2	声明继承	191
9.1.3	C#的继承原则	193
9.1.4	隐藏方法	196
9.2	多态	198
9.2.1	虚方法与重写方法	198
9.2.2	重写方法的性质	200
9.2.3	多态的魔力	203
9.3	关键字 base	206
9.3.1	base 访问	206
9.3.2	base 用于构造函数声明	207
9.4	抽象类与抽象方法	208
9.5	密封类与密封方法	211
9.6	静态类	213
9.7	接口	214
9.7.1	声明接口	214
9.7.2	接口实现与继承	217
9.7.3	显式接口成员	220
9.7.4	接口成员访问	224
9.7.5	接口与组件编程	228
9.8	所有类型的始祖：object	229
9.9	类的转换	233
9.9.1	隐式转换	234
9.9.2	显式转换	234
9.9.3	is 运算符	235
9.9.4	as 运算符	236

9.10	关于修饰符的总结	237
9.11	小结	239
9.12	练习	240
第 10 章	异常处理	243
10.1	处理异常	243
10.1.1	System.Exception	243
10.1.2	引发异常	244
10.1.3	处理异常	245
10.2	公共异常类	250
10.3	自定义异常类	253
10.4	改进自动取款机模拟程序：增加异常处理	254
10.5	小结	255
10.6	练习	256
第 11 章	重载运算符	257
11.1	重载运算符的原则和语法	257
11.2	重载标准运算符	258
11.2.1	重载一元运算符	259
11.2.2	重载二元运算符	261
11.3	重载转换运算符	264
11.4	小结	265
11.5	练习	265
第 12 章	结构与枚举	266
12.1	结构	266
12.1.1	定义结构	266
12.1.2	结构与类	269
12.2	枚举	273
12.2.1	声明枚举	273
12.2.2	枚举转换	275
12.3	小结	278
12.4	练习	278
第 13 章	集合	280
13.1	集合与接口	280
13.2	数组的高级功能	282
13.2.1	数组的方法和属性	282
13.2.2	CreateInstance 方法	286